CLAIM LISTING

Please amend claims 31, 42, 43, and 44 as indicated below.

1-30. (Canceled)

31. (Previously presented) A method, comprising:

creating a shadow cache configured to hold a**t least one** modified block that has been physically redone and undone or that has been logically undone;

collecting a**n active transaction** list of all active transactions when a read-only transaction starts;

determining a first log sequence number and a second log sequence number **associated with the active transaction list,** the first log sequence number and the second log sequence number **specifying a set of transactions capable of being physically redone and undone or logically undone**~~being used to compare another log sequence number of another modified block being loaded into the shadow cache~~;

loading a modified block into a read-only cache view, the modified block including a log sequence number;

comparing the log sequence number to both the first log sequence number and the second log sequence number;

performing physical redo **operations** or **physical** undo[[**ing**]] operations **based upon the comparing** as the ~~another~~ modified block is loaded into [[**a**]] **the** read-only cache view ~~as indicated by a comparison of the another log sequence number to the first log sequence number and the second log sequence number~~**, wherein the physical redo operations or the physical undo operations are not performed on the modified block when the log sequence number is less than the fist log sequence number, and the physical redo operations are performed on the modified block when the log sequence number is greater than the first log sequence number and less than the second log**

sequence number, and the physical undo operations are performed on the modified block when the log sequence number is greater than the second log sequence number;

processing ~~an~~ **the** active transaction list to physically undo an incomplete action and to logically undo an active transaction to generate transactional consistency in the read-only cache view;

creating one or more other modified blocks to generate the transactional consistency in the read-only cache view; and

determining **the** one or more other modified blocks ~~that were created to generate the transactional consistency in the read-only cache view, the one or more other modified blocks~~ being stored in the shadow cache and not an original database if a shared cache overflows.

32. (Previously presented) The method of claim 31, wherein the one or more other modified blocks are not written from the shadow cache to the original database.

33. (Previously presented) The method of claim 31, wherein the original database is used if the read-only cache view overflows a shared cache.

34. (Previously presented) The method of claim 31, further comprising using an allocation bitmap configured to indicate the modified block is to be temporarily stored in a temporary database.

35. (Previously presented) The method of claim 34, further comprising deleting the modified block temporarily stored in the shadow cache by updating the allocation bitmap, the updating being performed upon completing the read-only transaction.

36. (Previously presented) The method of claim 31, wherein determining one or more other modified blocks comprises mapping the one or more other modified blocks in a table in the shadow cache, the table comprising a first column configured to maintain a block number of a read-only cache view block having an undo/redo record applied to it and a second column

configured to maintain a block number allocated to temporarily store the one or more other modified blocks overflowing the shared cache and being stored in the shadow cache.

37. (Previously presented) The method of claim 31, further comprising marking the read-only cache view as closed upon termination of the read-only transaction.

38. (Previously presented) The method of claim 37, further comprising:

traversing the shared cache looking for a database block to purge when a new block allocation is needed in the shared cache; and

purging the database block from the read-only cache view that has been marked as closed.

39. (Previously presented) The method of claim 31, further comprising sharing the read-only cache view with other read-only transactions which start within a period of time following a start of the read-only transaction.

40. (Previously presented) The method of claim 31, further comprising:

detecting the read-only transaction; and

adding a back link log record to a transaction log, the back link log record being configured to link to another log record in the transaction log that is associated with a write transaction, the adding being performed upon occurrence of a write operation.

41. (Previously presented) The method of claim 40, further comprising using the back link log record to skip a portion of the transaction log that is irrelevant to undoing an uncommitted write transaction, wherein the back link log record is generated in the transaction log when there is an active read only transaction, if the read-only transaction must be undone.

42. (Currently amended) A computer-readable **storage** medium having processor-executable instructions for performing the method of claim 31.

43. (Previously presented) The method of claim 31, further comprising**:**

downloading a set of processor-executable instructions for performing the method of claim <u>3</u>1<u>**; and**</u>

<u>**storing the set of processor-executable instructions on a computer-readable storage medium configured to be accessed by at least one computer**</u>.

44. (Currently amended) A database system configured to restore a database to a consistent version supporting read-only uses, comprising:

a computer having a processor and memory;

a log manager module configured to manage a transaction log of the database system;

a cache manager module configured to manage a shared cache that stores one or more database blocks in a memory of the database system<u>,</u> and configured to create a write view of the database in the shared cache supporting read and write uses of the database and a read-only cache view of the database using the transaction log of the database, the read-only cache view being created in response to a read-only transaction of the database, the read-only cache view comprising the one or more database blocks of the shared cache that record a view of a version of the database at a time, wherein the cache manager is configured to use a non-transactional database to store the one or more database blocks that overflow the shared cache during use of the read-only cache view by the read-only transaction<u>**, wherein the steps to create the read-cache view include determining a first log sequence number and second log sequence number from the transaction log, loading a modified block including a log sequence number into the read-only cache view, comparing the log sequence number to the first log sequence number and the second log sequence number, and performing physical redo operations or physical undo operations based upon the comparing as the modified block is loaded into the read-only cache view, wherein the physical redo operations or the physical undo operations are not performed on the modified block when the log sequence number is less than the fist log sequence number, and the physical redo operations are performed on the modified block when the log sequence number is greater than the first log sequence number and less than the second log sequence number, and the physical undo operations are performed on the modified block when the log sequence number is greater than the second log sequence number**</u>; and

a transaction manager module configured to logically undo one or more transactions on demand, the one or more transactions having begun but ~~have~~ <u>**having**</u> yet to commit upon starting the read-only transaction in order to construct the read-only cache view comprising a transactionally consistent prior version of the database, performing the read-only transaction using the read-only cache view without blocking performance of the one or more transactions involving a write operation using a write view of the database, and returning a result associated with the read-only transaction.

45. (Previously presented) The system of claim 44, wherein a database block associated with the read-only cache view is not written from the shared cache to the given database during occurrence of the read-only transaction.

46. (Previously presented) The system of claim 44, wherein the cache manager stores a database block that overflows the shared cache in a temporary database.

47. (Previously presented) The system of claim 46, wherein the non-transactional database is used if the read-only cache view overflows the shared cache.

48. (Previously presented) The system of claim 44, wherein the cache manager maintains an allocation bitmap identifying the one or more database blocks being temporarily stored in a temporary database.

49. (Previously presented) The system of claim 48, wherein the cache manager is configured to delete the one or more blocks from the temporary database by updating the allocation bitmap.

50. (Previously presented) The system of claim 44, wherein the cache manager stores a mapping to the one or more blocks overflowing the shared cache in a table of the non-transactional database including a first column configured to maintain a block number of a read-only cache view block having an undo/redo record and a second column configured to maintain a block number allocated to temporarily store the one or more blocks overflowing the shared cache.

51. (Previously presented) The system of claim 44, wherein the cache manager is configured to mark the read-only cache view as closed upon termination of the read-only transaction.

52. (Previously presented) The system of claim 51, wherein the cache manager is configured to traverse the shared cache looking for the one or more blocks to purge, and is further configured to purge the one or more blocks from the read-only cache view if the one or more blocks from the read-only cache view have been marked as closed when a new block allocation is requested in the shared cache.

53. (Previously presented) The system of claim 44, wherein the cache manager is configured to share the read-only cache view created for the read-only transaction with other read-only transactions that are configured to start within a period of time following a start of the read-only transaction.

54. (Previously presented) The system of claim 44, wherein the log manager is configured to detect the read-only transaction, and further configured to add a back link log record to the transaction log that is configured to link together two or more log records of the transaction log that are associated with a write transaction to be logically undone.

55. (Previously presented) The system of claim 44, wherein the log manager is configured to use a back link log record to skip a portion of the transaction log that is not associated with undoing the write operation, wherein the back link log record is generated in the transaction log when there is an active read only transaction.